

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

Applicants:	Richard Bussiere et al.	
Serial No.:	10/713,560	
Filed:	November 14, 2003	
For:	DISTRIBUTED INTRUSION RESPONSE SYSTEM	
Assignee:	Enterasys Networks, Inc.	
Examiner:	Thomas M. Szymanski	
Art Unit:	2134	Confirmation No. 8242
		Paper No. 13

---

Commissioner for Patents  
P.O. Box 1450  
Alexandria, VA 22313-1450

**DECLARATION OF MARK TOWNSEND PURSUANT TO 37 CFR § 1.131**

Dear Sir:

In support of my claim of prior invention of the invention described in the referenced application in view of the Sung et al. reference cited in the June 29, 2007, office action, I hereby declare as follows:

1. My name is Mark Townsend. I am an applicant and co-inventor of the invention described and claimed in the referenced patent application.
2. In association with others, I conceived of the invention described in the application before April 14, 2003. Prior to January 2, 2003, I participated in discussions with one or more co-inventors regarding the implementation of the Distributed Intrusion Response System described in the Enterasys Invention Disclosure Form identified in the accompanying Declaration of Richard Graham.
3. Initially, the group of people involved in the discussions, including myself, undertook the task of developing the invention described in the Invention Disclosure Form during any spare time available to us beyond time spent carrying out our regular obligations to produce and market existing company products. As a result, we were not able to commit all of our time to continuous development of the invention. Nevertheless, we spent all available time diligently developing the invention. Our efforts included telephone conversations, e-mail exchanges, in-person meetings, tasking computer programmers to implement coding schemes, and reviewing code and evolving implementations of the invention. A copy of a representative first example of an e-mail exchange,

dated January 17, 2003, between myself and an Enterasys computer programmer I asked to provide programming services related to the development of the Distributed Intrusion Response System based on modifications to Enterasys's existing Dragon Intrusion Detection System (Dragon) and Enterasys's existing User Personal Network (UPN) network management system is attached hereto as Exhibit A.

4. Subsequent to January 17, 2003, I periodically communicated with one or more of the other co-inventors regarding means available to us to implement the invention as contemplated. I also communicated with others of Enterasys who I believed would be able to assist in the implementation through programming activities. One person I spoke with in particular, Andy Beats, was asked to explain the functioning of the Dragon and UPN operations and, in particular, we discussed means by which intrusion detection signals from the Dragon system could be trapped and transferred to the existing Enterasys network management system referred to as Atlas for the purpose of targeted network entry device modification. A copy of a second representative example of an e-mail exchange between myself and Mr. Beats, dated February 28, 2003, related to the implementation of the present invention is attached hereto as Exhibit B.

5. Subsequent to February 28, 2003, as I and other inventors performed our primary functions with the company, we continued to discuss the implementation of the invention and identified two Enterasys programmers with experience working with the Dragon system, Salo Fajer and Tom May, able and with some time available, to assist in coding the conceived implementation of the invention. Over the course of a few weeks, generated initial coding used to carry out the intended function of the invention. A copy of the preliminary coding for that purpose that either or both of Mr. Fajer and Mr. May provided to me for my review and dated April 15, 2003, is attached hereto as Exhibit C.

6. Subsequent to April 15, 2003, I worked with Mr. Fajer and Mr. May to complete the preliminary programming for the invention. A copy of a third example e-mail set generated over the period from May 1, 2003, to May 2, 2003, culminated in a statement by Mr. Fajer confirming his and Mr. May's efforts in the development of the programming for a demonstration model of the invention, all in response to internal discussions regarding improvements to network system security. A copy of that e-mail set is attached hereto as Exhibit D. Shortly thereafter, Mr. May and I discussed the generated computer program and certain features to be "tweaked." Mr. May

provided me a copy of his notes taken at the time of our discussion, specifically on May 7, 2003. A copy of Mr. May's notes as provided to me is attached hereto as Exhibit E.

7. Subsequent to May 7, 2003, over a period of several weeks during which all involved in the development of the invention continued to perform our regular duties for the company, Mr. May and Mr. Fajer, pursuant to my instructions and requests, made adjustments to the programming for the invention, until on or about July 28, 2007. On June 25, 2003, either or both of Mr. May and Mr. Fajer presented to me a summary flow diagram they had generated after substantially completing the coding for the demonstration model of the invention to perform as intended. That summary flow diagram of June 25, 2003, is attached hereto as Exhibit F.


8. Between June 25, 2003, and July 28, 2003, we completed final modifications to a working prototype of the invention. At that time, I began preparation of a PowerPoint® presentation for initiating interest in establishing the implementation of the invention as an official company development project. The summary flow diagram of the invention operating as intended was included and shown to company management. I completed the presentation on or about July 28, 2003. A copy of the presentation is attached hereto as Exhibit G.

9. On or about August 4, 2003, I reviewed completed Perl script programming code representing the implementation of the invention in a computing system. A copy of the code generated on or about August 4, 2003, is attached hereto as Exhibit H. I believe that the invention was operational for its intended purpose at least as of that date.

10. On or about August 6, 2003, I demonstrated to a prospective customer the invention as implemented. In the course of running an example of the invention on the Enterasys internal network system using the programming code of Exhibit H, I also showed to the prospective customer the presentation of Exhibit G.

11. Upon information and belief, the invention described in the referenced application was conceived at least as of January 2, 2003, and diligently reduced to practice no later than August 6, 2003.

12. I hereby declare that all statements made herein of my own knowledge are true and that all statements made on information and belief are believed to be true; and further that these statements were made with the knowledge that willful false statements and the like so made are punishable by fine or imprisonment, or both, under 18 USC 1001 and that such willful false statements may jeopardize the validity of the application or any patent issued.

By: \_\_\_\_\_

Mark Townsend, Applicant

Date: 10/25/07

**Townsend, Mark**

**From:** Skowronek, Kurt  
**Sent:** Friday, January 17, 2003 1:38 AM  
**To:** Townsend, Mark; Post, James  
**Subject:** RE: UPN/Dragon

Mark,

Jim and I worked on this yesterday afternoon and ran up against a couple of issues. The discovery script process would run the aliasgreb and statallas, we will need to do this in either windows or solaris. So we were moving down a path of having Dragon send a trap to console and console was going to initiate the "discovery script" and policy set. Using Console 1.1 is doable but needs more time to futz with it because we'll need to rewrite the dragon's trap into hp format so that it is recognizable by the alarm notification tool in Console.

After doing some digging and prodding, I was able to get a OLD copy(circa 2000) of linux based Q tools. I will try to play with them when I get a chance. If you know who can get us an updated linux tool set, this would make this script much easier to write and launch directly from Linux not involve the other mess.

I don't expect to have anything to you next week though. I am just resetting any expectation that you may have or have committed to.

I am extremely busy, trying to keep up with everything this month. I don't know why it is this busy so early in the quarter but I am booked out through until Feb already.  
<KJS>

-----Original Message-----

**From:** Townsend, Mark  
**Sent:** Monday, January 13, 2003 10:53 AM  
**To:** Post, James; Skowronek, Kurt  
**Subject:** UPN/Dragon

Gents,

Here's a sample slide regarding this morning's discussion.

If you have any thoughts, I'm all ears!

Thanks,

~Mark

<< File: Proof-of-Concept.ppt >>

Mark D. Townsend  
Reg. Director Systems Engineering  
Enterasys Networks  
50 Minuteman Road  
Andover, MA 01810  
Tel 978 684 1623  
[markt@enterasys.com](mailto:markt@enterasys.com)

Townsend, Mark

From: Townsend, Mark  
Sent: Friday, February 28, 2003 3:42 PM  
To: Beate, Andy  
Subject: thoughts 4 u

Importance: High

Andy,

Dragon 6 has a SNMP module on the sensor; you could have it, not alarm tool, send a SNMP trap to Atlas. I remember some of your old training from Durham -- let's use the tools in NetSight to set Policies for SNMP traps and take appropriate actions. I figured that would be old hat for you.

Just a thought.

~Mark

Mark D. Townsend  
Enterasys Networks  
978 684 1623

---SNIP---

Lab 6 (HIDS Active Response) is nowhere NEAR the point of STARTING development, because I will need either (1) to spend a couple of days with a development engineers who can provide me with solid example PERL scripts and how they can be used so that the field can benefit from the lab or (2) the Dragon Team will need to provide valid scripts and details on how to use them. **The goal here is to have the HIDS detect an event and kick off a PERL script (wrapped module) which will "do something in response to the detected event".** I need the team's advice on what that "something" might be, and an accompanying script that will accomplish the desired "active response".

Lab 7 (Alarmtool Active Response) has not even started development, as I do not want to use the suggested PERL script that adds an ACL to a Cisco router. I think it would be much better to modify the current script (or create a new one) that adds an ACL to an Enterasys product. I have not had time to work on that, and could very much use the team's assistance in this regard. [Sam Stover indicated that he can obtain a script which accomplishes this. If Sam's script is acceptable to the team, we'll put it into production for a lab exercise and run with it.]

## EXHIBIT

C

Role = Test UPN Test 01

Rule Kill TCP 80 assigned to VIO 99  
(discarded)

Frame Status Source Address Dest. Address Size Rel. Time Delta Time Abs. Time Summary  
 1196 THAY-XP1 [10.10.10.253] 160 0:04:02.358 0.006.402 04/15/2003 04:29:36 PM SHMP: Set enterprise.5624.1.2.6.1.5.1.3.5 .. enterprise.5

DLG: --- DLG Header ---

DLG:  
 DLG: Frame 1196 arrived at 16:29:36.5051; frame size is 160 (0x2) hex bytes.  
 DLG: Destination = station C01r1902F0  
 DLG: Source = station 000347854C8D  
 DLG: Ethertype = 0800 (IP)  
 DLG:

IP: --- IP Header ---

IP:  
 IP: Version = 4, header length = 20 bytes  
 IP: Type of service = 00  
 IP: 000. .... = routine  
 IP: .... 0. .... = normal delay  
 IP: .... 0. .... = normal throughput  
 IP: .... 0. .... = normal reliability  
 IP: .... 0. .... = ECN bit - transport protocol will ignore the ECN bit  
 IP: .... 0. .... = CS bit - no congestion  
 IP: Total length = 146 bytes  
 IP: Identification = 36504  
 IP: Flags = 0x  
 IP: .0. .... = may fragment  
 IP: .0. .... = last fragment  
 IP: Fragment offset = 0 bytes  
 IP: Time to live = 128 seconds/hops  
 IP: Protocol = 17 (UDP)  
 IP: Header checksum = 824F (correct)  
 IP: Source address = [10.10.10.99], THAY-XP1  
 IP: Destination address = [10.10.10.253]  
 IP: No options  
 IP:

UDP: --- UDP Header ---

UDP:  
 UDP: Source port = 1866  
 UDP: Destination port = 161 (SHMP)  
 UDP: Length = 126  
 UDP: Checksum = 0B2A (correct)  
 UDP: [118 byte(s) of data]  
 UDP:

SHMP: --- Simple Network Management Protocol (Version 1) ---

SHMP:  
 SHMP: SHMP Version = 1  
 SHMP: Community = public  
 SHMP: Command = Set request  
 SHMP: Request ID = 22112  
 SHMP: Error status = 0 (No error)  
 SHMP: Error index = 0  
 SHMP:  
 SHMP: Object = {1.3.6.1.4.1.5624.1.2.6.1.5.1.3.5} (enterprise.5624.1.2.6.1.5.1.3.5)  
 SHMP: Value = 4  
 SHMP:  
 SHMP: Object = {1.3.6.1.4.1.5624.1.2.6.1.5.1.2.5} (enterprise.5624.1.2.6.1.5.1.2.5)  
 SHMP: Value = Test  
 SHMP:  
 SHMP: Object = {1.3.6.1.4.1.5624.1.2.6.1.5.1.4.5} (enterprise.5624.1.2.6.1.5.1.4.5)  
 SHMP: Value = 2  
 SHMP:  
 SHMP: Object = {1.3.6.1.4.1.5624.1.2.6.1.5.1.6.5} (enterprise.5624.1.2.6.1.5.1.6.5)  
 SHMP: Value = 2  
 SHMP:

Frame Status Source Address Dest. Address Size Rel. Time Delta Time Abs. Time Summary  
 1197 [10.10.10.253] THAY-XP1 160 0:04:02.374 0.015.501 04/15/2003 04:29:36 PM SHMP: GetReply enterprise.5624.1.2.6.1.5.1.3.5 .. enterprise.5

DLG: --- DLG Header ---

DLG:  
 DLG: Frame 1197 arrived at 16:29:36.5206; frame size is 160 (0x0A) hex bytes.

Note: Discard VIO = 4000.

(Roll) Test 01

L TCP 999 ↓

L ETH IP 99 ↓

Role Test R

UPN

Test 02

144

149

393 → 398

(Roll) Test 01

L mac "99"

150 → 154

UPN

Test 03

144

Policy Profile  
table

7 (what is 57)

) group? #

) Role name!

DLC: Destination = Station 000347854C8D  
DLC: Source = Station Chlcr19C02F0  
DLC: Ethertype = 0800 (IP)  
DLC:

IP: ----- IP Header -----

IP:  
IP: Version = 4, header length = 20 bytes  
IP: Type of service = 00  
IP: 000. .... = routine  
IP: ....0 .... = normal delay  
IP: .... 0... = normal throughput  
IP: .... .0.. = normal reliability  
IP: .... ..0 = RCT bit - transport protocol will ignore the CE bit  
IP: .... ..0 = CE bit - no congestion  
IP: Total length = 146 bytes  
IP: Identification = 666  
IP: Flags = 0x  
IP: .0.. .... = may fragment  
IP: ..0. .... = last fragment  
IP: Fragment offset = 0 bytes  
IP: Time to live = 255 seconds/hops  
IP: Protocol = 17 (UDP)  
IP: Header checksum = 8F4D (correct)  
IP: Source address = [10.10.10.253]  
IP: Destination address = [10.10.10.99], TMAN-XPI  
IP: No options  
IP:

UDP: ----- UDP Header -----

UDP:  
UDP: Source port = 161 (SNMP)  
UDP: Destination port = 1996  
UDP: Length = 126  
UDP: No checksum  
UDP: [119 byte(s) of data]

UDP:

SNMP: ----- Simple Network Management Protocol (Version 1) -----

SNMP:  
SNMP: SNMP Version = 1  
SNMP: Community = public  
SNMP: Command = Get response  
SNMP: Request ID = 22112  
SNMP: Error status = 0 (No error)  
SNMP: Error index = 0  
SNMP:  
SNMP: Object = {1.3.6.1.4.1.5624.1.2.6.1.5.1.3.5} (enterprise.5624.1.2.6.1.5.1.3.5)  
SNMP: Value = 4  
SNMP:  
SNMP: Object = {1.3.6.1.4.1.5624.1.2.6.1.5.1.2.5} (enterprise.5624.1.2.6.1.5.1.2.5)  
SNMP: Value = Test  
SNMP:  
SNMP: Object = {1.3.6.1.4.1.5624.1.2.6.1.5.1.4.5} (enterprise.5624.1.2.6.1.5.1.4.5)  
SNMP: Value = 2  
SNMP:  
SNMP: Object = {1.3.6.1.4.1.5624.1.2.6.1.5.1.6.5} (enterprise.5624.1.2.6.1.5.1.6.5)  
SNMP: Value = 2  
SNMP:

Frame Status	Source Address	Dest. Address	Size	Rel. Time	Delta Time	Abs. Time	Summary
1196	TMAN-XPI	[10.10.10.253]	133	0:04:02.378	0.004.403	04/15/2003 04:29:36 PM	SNMP: Set enterprise.5624.1.2.6.2.4.1.3.5.12, enterprise.1

DLC: ----- DLC Header -----

DLC:  
DLC: Frame 1196 arrived at 16:29:36.5251; frame size is 133.(0085 hex) bytes.  
DLC: Destination = Station Chlcr19C02F0  
DLC: Source = Station 000347854C8D  
DLC: Ethertype = 0800 (IP)  
DLC:

IP: ----- IP Header -----

IP:  
IP: Version = 4, header length = 20 bytes



IP: Type of service = 00  
 IP: 000 .... = routine  
 IP: ...0 .... = normal delay  
 IP: .... 0... = normal throughput  
 IP: .... .0.. = normal reliability  
 IP: .... ..0 = ECF bit - transport protocol will ignore the CE bit  
 IP: .... ...0 = CE bit - no congestion  
 IP: Total length = 119 bytes  
 IP: Identification = 36505  
 IP: Flags = 0X  
 IP: ..0 .... = may fragment  
 IP: ...0 .... = last fragment  
 IP: Fragment offset = 0 bytes  
 IP: Time to live = 128 seconds/hops  
 IP: Protocol = 17 (UDP)  
 IP: Header checksum = 8269 (correct)  
 IP: Source address = [10.10.10.99], TRAY-XPI  
 IP: Destination address = [10.10.10.253]  
 IP: No options  
 IP:

UDP: ---- UDP Header ----

UDP:  
 UDP: Source port = 1866  
 UDP: Destination port = 161 (SNMP)  
 UDP: Length = 99  
 UDP: Checksum = 25F7 (correct)  
 UDP: [91 byte(s) of data]  
 UDP:

SNMP: ---- Simple Network Management Protocol (Version 1) ----

SNMP:  
 SNMP: SNMP Version = 1  
 SNMP: Community = public  
 SNMP: Command = Set request  
 SNMP: Request ID = 22113  
 SNMP: Error status = 0 (No error)  
 SNMP: Error index = 0  
 SNMP:  
 SNMP: Object = [1.3.6.1.4.1.5624.1.2.6.2.4.1.3.5.12] (enterprise.5624.1.2.6.2.4.1.3.5.12)  
 SNMP: Value = 4  
 SNMP:  
 SNMP: Object = [1.3.6.1.4.1.5624.1.2.6.2.4.1.2.5.12] (enterprise.5624.1.2.6.2.4.1.2.5.12)  
 SNMP: Value = [1.3.6.1.4.1.52.4.1.2.16.6.1.4.1.5.99.18.88.0]  
 SNMP:

Frame	Status	Source Address	Dest. Address	Size	Rel. Time	Delta Time	Abs. Time	Summary
1199		[10.10.10.253]	TRAY-XPI	133	0:04:02.393	0.014.650	04/15/2003 04:29:36 PM	SNMP: GetReply enterprise.5624.1.2.6.2.4.1.3.5.12, enterprise.1

DLC: ---- DLC Header ----

DLC:  
 DLC: Frame 1199 arrived at 16:29:36.5397; frame size is 133 (0085 hex) bytes.  
 DLC: Destination = Station 000347B54C8D  
 DLC: Source = Station Cb1r19C02F0  
 DLC: Ethernertype = 0800 (IP)  
 DLC:

IP: ---- IP Header ----

IP:  
 IP: Version = 4, header length = 20 bytes  
 IP: Type of service = 00  
 IP: 000 .... = routine  
 IP: ...0 .... = normal delay  
 IP: .... 0... = normal throughput  
 IP: .... .0.. = normal reliability  
 IP: .... ..0 = ECF bit - transport protocol will ignore the CE bit  
 IP: .... ...0 = CE bit - no congestion  
 IP: Total length = 119 bytes  
 IP: Identification = 667  
 IP: Flags = 0X  
 IP: ..0 .... = may fragment  
 IP: ...0 .... = last fragment  
 IP: Fragment offset = 0 bytes

IP: Time to live = 255 seconds/hops  
IP: Protocol = 17 (UDP)  
IP: Header checksum = 8f67 (correct)  
IP: Source address = [10.10.10.253]  
IP: Destination address = [10.10.10.99], TNAV-KPI  
IP: No options  
IP:

UDP: ----- UDP Header -----

UDP:  
UDP: Source port = 161 (SNMP)  
UDP: Destination port = 1966  
UDP: Length = 99  
UDP: No checksum  
UDP: [31 byte(s) of data]  
UDP:

SNMP: ----- Simple Network Management Protocol (Version 1) -----

SNMP:  
SNMP: SNMP Version = 1  
SNMP: Community = public  
SNMP: Command = Get response  
SNMP: Request ID = 22113  
SNMP: Error status = 0 (No error)  
SNMP: Error index = 0  
SNMP:  
SNMP: Object = [1.3.6.1.4.1.5624.1.2.6.2.4.1.3.5.12] (enterprise.5624.1.2.6.2.4.1.3.5.12)  
SNMP: Value = 4  
SNMP:  
SNMP: Object = [1.3.6.1.4.1.5624.1.2.6.2.4.1.2.5.12] (enterprise.5624.1.2.6.2.4.1.2.5.12)  
SNMP: Value = [1.3.6.1.4.1.52.4.1.2.16.6.1.4.1.5.99.10.00.0]  
SNMP:

Frame Status	Source Address	Dest. Address	Source Port	Delta Time	Rel. Time	Summary
1200	TNAV-KPI	[10.10.10.253]	90	0:04:02.400	0.006.832	04/15/2003 04:29:36 PM SNMP: Set

Cabletron.4.1.2.14.7.1.1.0 - 2

DLCL: ----- DLCL Header -----

DLCL:  
DLCL: Frame 1200 arrived at 16:29:36.5466; frame size is 90 (005A hex) bytes.  
DLCL: Destination = Station Chtr19C02F0  
DLCL: Source = Station 000347B54C8D  
DLCL: Ethertype = 0800 (IP)  
DLCL:

IP: ----- IP Header -----

IP:  
IP: Version = 4, header length = 20 bytes  
IP: Type of service = 00  
IP: 000. .... = routine  
IP: .... 0 .... = normal delay  
IP: .... 0 .... = normal throughput  
IP: .... 0 .... = normal reliability  
IP: .... 0 .... = ECN bit - transport protocol will ignore the CE bit  
IP: .... 0 .... = CE bit - no congestion  
IP: Total length = 76 bytes  
IP: Identification = 36506  
IP: Flags = 0x  
IP: .... 0 .... = may fragment  
IP: .... 0 .... = last fragment  
IP: Fragment offset = 0 bytes  
IP: Time to live = 120 seconds/hops  
IP: Protocol = 17 (UDP)  
IP: Header checksum = 8293 (correct)  
IP: Source address = [10.10.10.99], TNAV-KPI  
IP: Destination address = [10.10.10.253]  
IP: No options  
IP:

UDP: ----- UDP Header -----

UDP:  
UDP: Source port = 1866  
UDP: Destination port = 161 (SNMP)  
UDP: Length = 56  
UDP: Checksum = 420C (correct)

UDP: [48 byte(s) of data]  
UDP:  
SNMP: ----- Simple Network Management Protocol (Version 1) -----

SNMP:  
SNMP: SNMP Version = 1  
SNMP: Community = public  
SNMP: Command = Set request  
SNMP: Request ID = 22114  
SNMP: Error status = 0 (No error)  
SNMP: Error index = 0  
SNMP:  
SNMP: Object = {1.3.6.1.4.1.52.4.1.2.14.7.1.1.0} (Cabletron.4.1.2.14.7.1.1.0)  
SNMP: Value = 2  
SNMP:

Frame Status	Source Address	Dest. Address	Size	Rel. Time	Delta Time	Ans. Time	Summary
1201	[10.10.10.253]	THAY-XP1	59	0:04:02.497	0.007.496	04/15/2003 04:29:36 PM	SNMP: GetReply Cabletron.4.1.2.14.7.1.1.0 = 2

DLC: ----- DLC Header -----

DLC:  
DLC: Frame 1201 arrived at 16:29:36.5541; frame size is 50 (005A hex) bytes.  
DLC: Destination = Station 00037B54C8D  
DLC: Source = Station Cbltr19C02F0  
DLC: Ethertype = 0800 (IP)  
DLC:

IP: ----- IP Header -----

IP:  
IP: Version = 4, header length = 20 bytes  
IP: Type of service = 00  
IP: 000. .... = routine  
IP: .... 0 .... = normal delay  
IP: .... 0... = normal throughput  
IP: .... .0. = normal reliability  
IP: .... .0. = ECN bit - transport protocol will ignore the CE bit  
IP: .... .0. = CE bit - no congestion  
IP: Total length = 76 bytes  
IP: Identification = 669  
IP: Flags = OK  
IP: .0. .... = may fragment  
IP: ..0. .... = last fragment  
IP: Fragment offset = 0 bytes  
IP: Time to live = 255 seconds/hops  
IP: Protocol = 17 (UDP)  
IP: Header checksum = 8F91 (correct)  
IP: Source address = [10.10.10.253]  
IP: Destination address = [10.10.10.99], THAY-XP1  
IP: No options  
IP:

UDP: ----- UDP Header -----

UDP:  
UDP: Source port = 161 (SNMP)  
UDP: Destination port = 1866  
UDP: Length = 56  
UDP: No checksum  
UDP: [48 byte(s) of data]  
UDP:

SNMP: ----- Simple Network Management Protocol (Version 1) -----

SNMP:  
SNMP: SNMP Version = 1  
SNMP: Community = public  
SNMP: Command = Get response  
SNMP: Request ID = 22114  
SNMP: Error status = 0 (No error)  
SNMP: Error index = 0  
SNMP:  
SNMP: Object = {1.3.6.1.4.1.52.4.1.2.14.7.1.1.0} (Cabletron.4.1.2.14.7.1.1.0)  
SNMP: Value = 2  
SNMP:

---

**Townsend, Mark**

**From:** Fajer, Salo  
**Sent:** Friday, May 02, 2003 11:30 AM  
**To:** Dragon Team  
**Subject:** RE: IPS future . . . and Enterasys UPN

I agree that there is a synergy between UPN and Dragon that we could take advantage of in the IPS story. Tom May and I are building a demo to use Dragon events to automatically set UPN Policies. We would love to see us integrate the existing tools we have to offer LAN based per port IPS! I know this has been discussed before, but imagine that Dragon sees a threat from an internal IP. It launches a Compass search and identifies the port in the network of the offender. Based on a previously set policy, it sets that port into a "penalty box" role and alerts the network admin. Or in a more advanced mode, blocks just that tcp port (i.e. SQL Slammer) on the physical port or denies ICMP. We have the current tools and features. It just needs to be tied together.

Unfortunately, the customer may purchase another vendor for the gateway device for now. However, we can actively secure the rest of the LAN.

On a less ambitious level of integration, the new N series are soon supposed to be able to trap on a hit on a policy or track application level usage. We need to make sure that the Matrix team sends this information out (i.e. syslog) in a method that Dragon can read. And, we need to make sure that we have a signature set to read this so that we at least merge UPN security events into Realtime.

What do you all think?

Thanks  
Salo

-----Original Message-----

**From:** Beats, Andy  
**Sent:** Friday, May 02, 2003 8:16 AM  
**To:** Marsh, Todd; Savage, David; Lau, Ed; Hamilton, Jane; Fernandes, Cliff  
**Cc:** Dragon Team  
**Subject:** RE: IPS future . . . and Enterasys UPN  
**Importance:** High

Todd, I don't argue the point you made. But with all due respect, don't we (ETS) need to be out there convincing our customers of what they need? Sure, customers know their networks and they should be telling us what they need. But don't we have an opportunity to cajole, market, sell, etc. them on the Enterasys solutions? Hell, Cisco did it, all the way to the number one spot.

I understand IPS is a hot topic. But Dragon and UPN already exist, and have an extensive history to boot. With some savvy positioning, can't we elevate this solution in the customers' eyes? Shouldn't we break the paradigm that IPS is the end-all/be-all in this regard, and slide the ETS solutions in there at all possible opportunities to do so?

Maybe ETS stock will continue its rise even faster. ☺

My two cents . . .

-Andy

-----Original Message-----

**From:** Marsh, Todd  
**Sent:** Friday, May 02, 2003 6:20 AM

**To:** Beats, Andy; Savage, David; Lau, Ed; Hamilton, Jane; Fernandes, Cliff  
**Cc:** Dragon Team  
**Subject:** RE: IPS future . . . and Enterasys UPN

In the summer time frame we will start to see some basic abilities in our Xpedition platform (e.g. the ability of the router to recognize that the flow set up rates are approaching a critical point and dynamically rate limit the line card that is causing the rise).

The UPN Acceptable Use Policy is fine but it is not dynamic and that is what customers are looking for.

-----Original Message-----

**From:** Beats, Andy  
**Sent:** Thursday, May 01, 2003 7:00 PM  
**To:** Savage, David; Lau, Ed; Hamilton, Jane; Fernandes, Cliff  
**Cc:** Dragon Team  
**Subject:** RE: IPS future . . . and Enterasys UPN  
**Importance:** High

Folks, let's not omit the fact that our UPN strategy includes a "Threat Management" service. It's not exactly Intrusion Prevention, but it aligns our customers OUTSTANDINGLY WELL with the overall Enterasys security solution.

-----Original Message-----

**From:** Savage, David  
**Sent:** Thursday, May 01, 2003 5:34 PM  
**To:** Lau, Ed; Hamilton, Jane; Fernandes, Cliff  
**Cc:** Dragon Team  
**Subject:** RE: IPS future

I think we should discuss this on our next Dragon Team call.

-----Original Message-----

**From:** Lau, Ed  
**Sent:** Thursday, May 01, 2003 4:14 PM  
**To:** Savage, David; Hamilton, Jane; Fernandes, Cliff  
**Cc:** Dragon Team  
**Subject:** RE: IPS future

Well said – even if we can roadmap it, it would go a long way towards keeping/winning customers.

Edward Lau  
Systems Engineer  
Enterasys Networks  
(212) 946-9279  
(212) 643-9587 Fax

-----Original Message-----

**From:** Savage, David  
**Sent:** Thursday, May 01, 2003 3:22 PM  
**To:** Hamilton, Jane; Fernandes, Cliff  
**Cc:** Dragon Team  
**Subject:** RE: IPS future

I think we need to look at today. Our competitors are starting to eat away at us with their NIDS IPS stories.

-----Original Message-----

**From:** Hamilton, Jane  
**Sent:** Thursday, May 01, 2003 1:00 PM  
**To:** Fernandes, Cliff  
**Cc:** Dragon Team  
**Subject:** RE: IPS future

We haven't really had a chance to evaluate it but may be looking at it in the near future.

-----Original Message-----

**From:** Fernandes, Cliff  
**Sent:** Thursday, May 01, 2003 1:53 PM  
**To:** Hamilton, Jane  
**Cc:** Dragon Team  
**Subject:** RE: IPS future

Thank you Jane. I will continue to use those avenues. There was a partner in China or Japan that had worked on an IPS for/with us a while back and we were going to test that internally. Is that still ongoing or did it not pan out?

Thanks  
Cliff

-----Original Message-----

**From:** Hamilton, Jane  
**Sent:** Thursday, May 01, 2003 1:43 PM  
**To:** Fernandes, Cliff; Dragon Team  
**Subject:** RE: IPS future

We address Host IPS in the roadmap presentation on the intranet site. We also discuss Active response, our current solution for Network IPS. Further review of network IPS is being discussed now as part of our 7.0 planning. Plus we in the process with Toronto of developing a Router, FW, VPN, limited IDS device that is planned to provide Network IPS in a couple of ways. That is most likely included on the XSR roadmap.  
Regards,  
Jane

-----Original Message-----

**From:** Fernandes, Cliff  
**Sent:** Thursday, May 01, 2003 1:17 PM  
**To:** Dragon Team  
**Subject:** IPS future

Hello, I have a customer looking for information regarding our IPS strategy. Do we have anything that a customer under NDA can have access to for thier security planning?  
Thanks

**Cliff Fernandes**

5/7

exVlanClassify IngressList *Instance*  
 4000.4.1.0  
 4000.15.999.0  
 4000.18.23.0  
 4000.18.90.0  
 4000.24.0.65536

taken from RH262 w/

static vlans 1 default  
 4000 discard

Vlan Classification Configurations

4000 IP Port type ICMP  
 4000 R/L WIP Port 999  
 4000 R/L TSP Port 23 telnet  
 4000 R/L TCP Port 80 HTTP  
 4000 R/L MAC address mac: 00000001

Then clearing NVRAM I configured:

- static VIO = 4000 w/NO egress ports = discard  
 R/L TCP 80 40 VIO 4000

Given:

	Row	Formatted Value
IngressList	4000.18.80.0	00.00.00.00 0-0 0-0 0-0
PortStatus		
Row Info		

Then I configured Port Configurations

under the Rule → activate on Port #1  
 Notice that as I set the ports, a 1 bit is set  
 in the "Formatted" field

# cxVlanClassifyTgressList

Formatted Value

position → 1 2 3 4 5 6 7 8 1 2 3 4 5 6 7 8 1 2 3 4 5 6 7 8 1  
 Port or / off 0/1 < → 0/1  
 example: 1 1 1 0 1 0 0 0 1 1 1 0 1 0 0 0 1 1 1 0 1 0 0 0 0  
 Hex → ES ES ES : 0  
 ↳ = Raw Data.

## Process

1. Create discard Vlan
  2. Set Rule
  3. enable Rule
- from MIB → dot1dBridge → gbridge MIB → MIB2 → data
- ① dot1d Vlan Editd. instance Run  
 1.3.6.1.2.1.17.7.1.4.2.1.1 [4000] [Discard]
- ② cxVlanClassifyTgressList  
 1.3.6.1.4.1.50.1.2.16.6.1.4.1.5 [4000, 18, 80, 0] [ES, ES, 1]
- ③ cxVlanClassifyRawStatus  
 1.3.6.1.4.1.6 [4000, 18, 80, 0] [1]

EOF



Policy changes ... "otssys Policy Profile m/b"

Policy Box Policy ...

- Post Policy Profile  $\equiv$  Role to post  
instance  $\equiv$  as its

Raw Value  $\equiv$  as Policy number!

→ "Post Policy Profile Admin ID" →  
1.3.1.4.1.5024.1.2.6.3.4

Name

Policy Profile Name 5024.1.2.6.1

instance  $\equiv$  Policy number

Raw Value  $\equiv$  Name

Policy Profile Post Vid

instance  $\equiv$  Policy #

Raw Value  $\equiv$  VID #

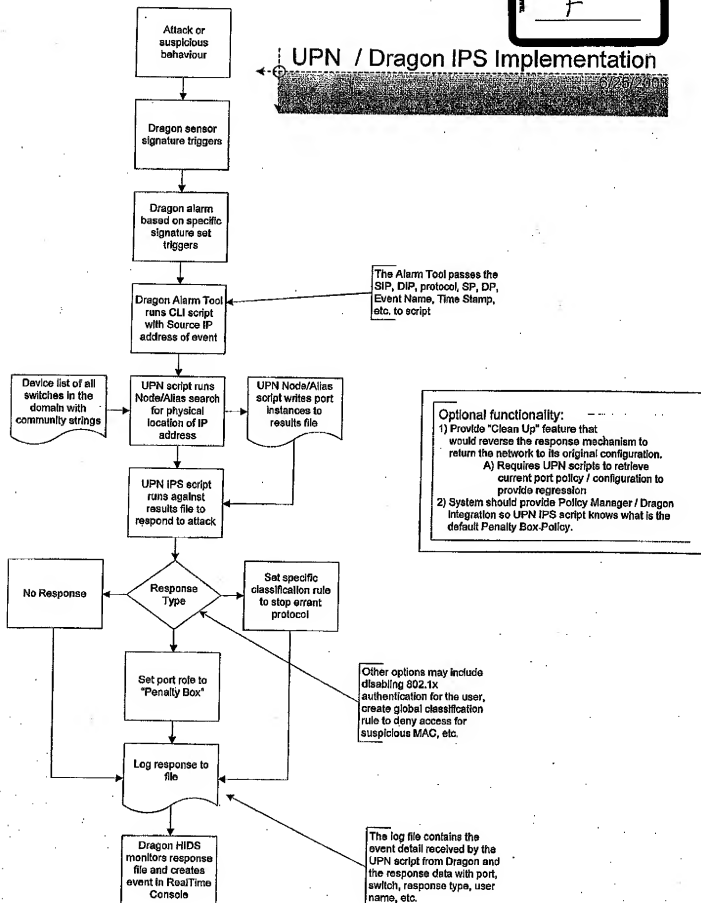
Policy Profile Post Vid Status

instance  $\equiv$  Policy #

Raw  $\equiv$  enabled/disabled 1/1/2

## UPN / Dragon IPS Implementation

6/26/2005





# IDS Integration with Intelligent LAN

## A Secure Networks Proposal

ENTERASYS NETWORKS PROPRIETARY & CONFIDENTIAL INFORMATION

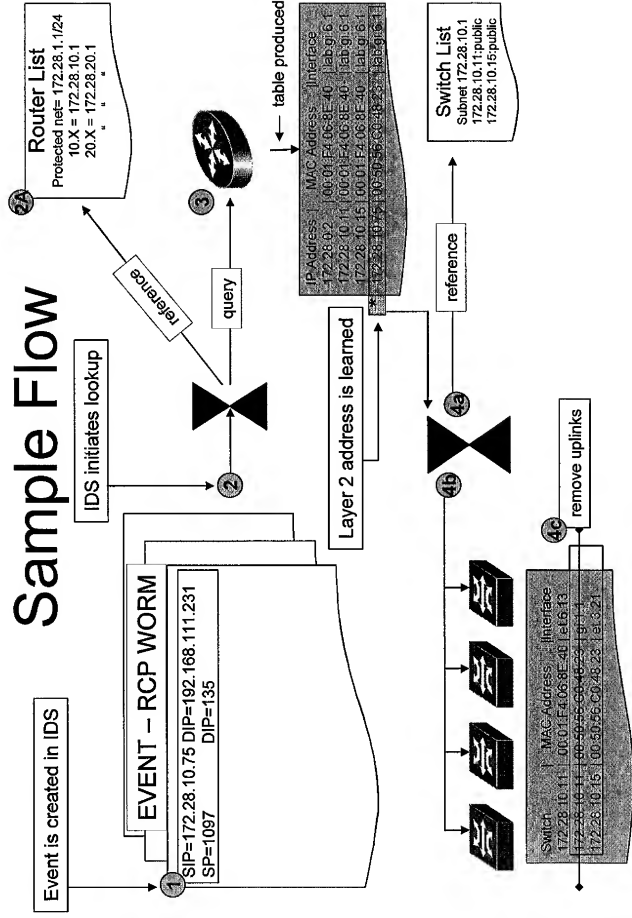
## 3 Separate Functions

- “Learn” – Physical ingress of event.
- “Log” – integrate learned “location” information and store with security event data.
- “Respond” – Perform logical change.

# Learn

- Process in which security management system can associate a security event with it's physical ingress point to the network
  - First step before any 'action' (mitigation) can occur.
  - Multiple methods to do this.
    - Initiated by IDS
      - IDS does lookup (locator process)
      - IDS passes information to Management for lookup

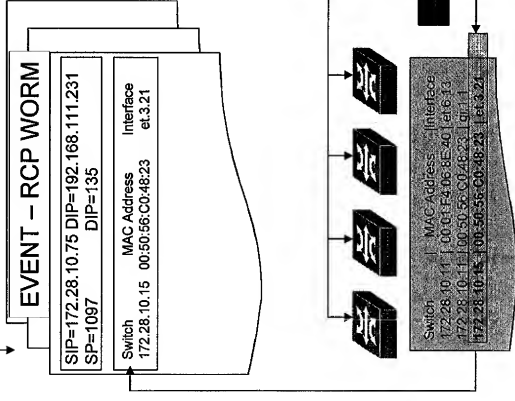
# Sample Flow



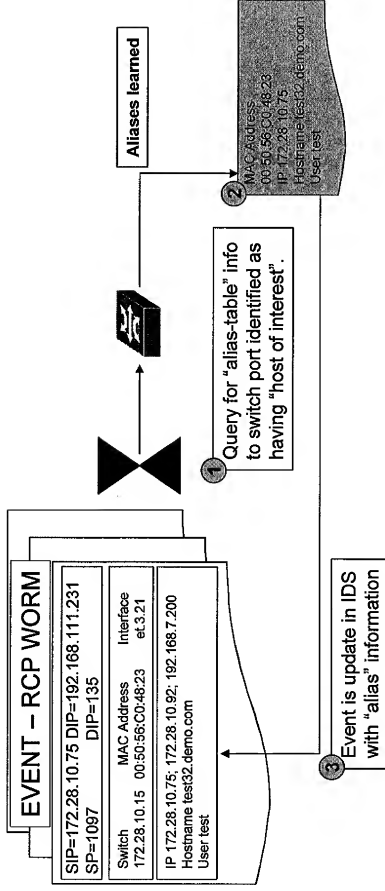
5

Event is update in IDS  
with "location" information

# Sample Flow



# Optional – Alias Correlation





# LOG

- Addition of “location” adds a new variable to the Security Database for reference
- Useful for historical correlation of security events
- Useful for tuning of “AUP”

Dragon 6 Server - Microsoft Internet Explorer

File Edit View Favorites Tools Help

Address <https://dragon/dragon/>

DRAGON

Trending Event Summary IP Address Summary Event Detail Additional Reports

Realtime Forensics Trending Reporting /

Policy Manager Asymptot

apply reset

# Events Details # Unique IPs Most Active IPs

Date	Event Name	# Events	Details	# Unique IPs	Most Active IPs
2003-07-28	ICMP-SUPERSCAN	511	256	134.141.110.136	
2003-07-28	SNMP-PUBLIC	256	255	134.141.110.136	
2003-07-28	TCP-SCAN	9	9	134.141.110.136	
2003-07-28	SSH-VERSION-2	3	4	172.28.1.2	
2003-07-28	SSH-VERSION-1	3	3	172.28.1.15	

Local intranet

Traditional DB contains what can be collected from the packet

- SIP / DIP (Layer 3)
- SP / DP (Layer 4)
- Event type
- # Events

Queries are limited to what data is in the DB!

ENTERASYS NETWORKS PROPRIETARY & CONFIDENTIAL INFORMATION

Dragon 6 Server - Microsoft Internet Explorer

File Edit View Favorites Tools Help

Address <https://dragon/dragon/>

DRAGON PolicyManager Alarms

Trending Event Summary IP Address Summary Event Detail Additional Reports

sensors date hosts events

Realtime Forensics Trending Reporting PolicyManager Alarms

apply reset

Date/Time	Source IP	Destination IP	Event Name	Location	Port
2003-07-28 05:13:07	172.28.10.75	172.28.1.1	RPC: WORM	172.28.10.15	Interface 3.21
2003-07-28 07:48:03	172.28.10.75	172.28.7.28	SNMP-PUBLIC	172.28.10.15	Interface 3.21
2003-07-28 17:13:07	172.28.1.1	134.141.110.136	PROXY:WEB-GET	172.28.10.15	Interface 3.21
2003-07-28 18:13:07	172.28.10.75	172.28.10.60	SNMP-PUBLIC	172.28.10.15	Interface 3.21
2003-08-28 03:12:05	172.28.10.18	172.28.1.61	SSH-VERSION-1	172.28.10.15	Interface 3.21

Local Intranet

Adding "Location" information to the same DB provides the admin a look at the Security Events from a different angle. In this scenario, the switch port is in a conference room. Implementing "Acceptable Use Policy" would have prevented 3 of above events from ever being forwarded.

ENTERASYS NETWORKS PROPRIETARY & CONFIDENTIAL INFORMATION

# RESPOND

- Knowing “Location” allows the IT Admin to program the system with “automated responses”
  - Type of Response is relevant to the “threat”.
- Scenario:
  - User attempts to gain “root” access to a server that he has permissions to (FTP).
    - Dragon is configured to monitor for “root” access
      - Learns Location
      - Automated Policy Change to Location (Port) to deny FTP services
  - Net affect – Proactive Security of resources
    - (Auditing of AUP)

## FinalUPN02

```
#!/usr/bin/perl

# Mod date 8/4/03
# New:
# Features:
# Input from CLI: [UserIP, SwitchIP, comm string, protocol and port]
# Output: [Interface #, MAC, TargetIP, switchIP] into logfile
# And SNMP set of rule to kill the service...
#.....

($xTargetIP, $DestIP, $Proto, $Port, $EventName, $Sensor) = @ARGV;
$xSwitchIP = "10.10.10.252"; # static setting for test... vs. reading a file
$xCommString = "public";
$response = 'UPN-SERVICE';
if ($Proto == 0) { $xProto = 'TCP'; }

# Here we remove the (.)s from the TargetIP data....
$x = '0';
for ($_ = $xTargetIP;
s/\./ /; ) {
    $x = $x + 1;
    $IP = $_;
    if ($x == 3)
    { $TargetIP = $IP;
    }
}
# End TargetIP cleanup.....

#.....
# Section 0, Var and file definitions
#.....

# Switch IP address = $xSwitchIP..... Switch address(s)[may be multiple]
# Switch Community = $xCommString..... Community string
# Target IP address = $xTargetIP..... The user you are looking for
# Node Alias file = nodealias.txt..... initial info return from switch
# Cleaned Alias file = UserInfo.txt..... cleaned nodealias.txt
# Final Log file = DragonLog.txt..... IP/mac/switch/name/port
# Protocol TCP/UDP = $Proto
# Port(s) = $Port

#.....
# Section 1, get node alias table
#.....

#$SwitchIP = $xSwitchIP;
use Net::SNMP(qw(snmplib snmp_event_loop oid_lex_sort));
use vars qw($session $error $result $sysupTime $tableloop $response $object);
$result = 'null';

($session, $error) = Net::SNMP->session(
    -hostname => $xSwitchIP,
    -community => $xCommString,
    -port => 161
);

if (!defined($session)) {
    printf("Error1: %s.\n", $error);
    exit 1;
}

$object = "1.3.6.1.4.1.52.4.1.3.7.1.1.1";
# This is the Node Alias OID ....
```

```
# $object = "1.3.6.1.4.1.5624.1.2.6.1.4.0"; Policy MIB, next avail policy index
```

```
#... print "\n";
$result = $session->get_table($object);
if (defined($result)) {
    printf ("Error2 on get request: %s.\n", $session->error);
}
```

```
# Here we need to loop through the table that we just read.
```

```
$table = @_;
```

```
$next = "null";
foreach $oid (oid_lex_sort(keys(%{$session->var_bind_list})))
{
    if ($object ne $oid) {
        $next = $oid;
        $table->{$oid} = $session->var_bind_list->{$oid};
    }
    else
    {
        $next = undef;
        last;
    }
}
```

```
# Here we copy the output to a text file...;
open (OUT, '>nodealias.txt') or die "couldn't open text file...\n";
```

```
foreach $oid (oid_lex_sort(keys(%{$table}))) {
    if($oid ne ""){
        print OUT ("{$oid},{$table->{$oid}}\n");
        # X printf("%s %s\n", $oid, $table->{$oid});
    } #end if
} # end foreach
```

```
$session->close;
close (OUT);
```

```
#.....
# Section 2, Search for the IP address and get the hash code, clean up.
#.....
```

```
# USING HASH CODE found in the table...
# Our list to hold the alias table...;
# X print "Target user IP is ... $xTargetIP \n";
```

```
open (HASHIN, "nodealias.txt") or die "where's the file...?\n";
while(<HASHIN) {
    $TheRec = $_;
    if($TheRec ne "") {
        chomp($TheRec);
        ($key, $val) = split(/,/, $TheRec, 2);
        # This splits our alias table into two parts, key and value
        if(defined($val)){ # is the $val wacky?
            if($val eq $TargetIP)
                # Here we search on the IP... [$TargetIP]
            {
                open (ALIASTABLE, '>cleaned.txt') or die "No file...\n";
                @newval = ($key);
                $i = '0';
                # X print "The variable is $key ... \n\n";
                for ($_ = $key; s/\./,/;){
                    [# Here we replace the . with a comma (,)...
                }
            }
        }
    }
}
```

# FinalUPN02

```

        $i = $i + 1;
        $FinalSort = $_;
        @HASH = ($FinalSort);
    }
} # end if statement
else {
    # print "Jumped to the else side... oops...\n\n";
} # end else
    } # end if
    } # end if
    } # end while

#
# Next get hash code from FinalSort elements 17,18...
# Then you need to split the one item string into many strings /\ ...!

@Hash = split(/,/,$FinalSort);
$Hashnumber1 = $Hash[16];
$Hashnumber2 = $Hash[17];

# Now you have the Hash variables...

close (HASHIN);
close (ALIASTABLE);

#... system('rm -f UserInfo.txt');

# Now we use the hash code to search the original nodealias file to get
# the port/mac/name of our target IP...
# So we re-open the nodealias file...

@Hash = ();
open (RESORT, 'nodealias.txt') or die "No file found...\n";
while(<RESORT>) {
    $n = '0';
    $Line = $_;
    chomp($Line);
    if ($Line ne "") {
        ($key1, $val1) = split(/,/, $Line, 2);
        for ($_ = $key1; s/\./,/;){
            $Final = $_;
            @Hash = ($Final);
            $n = $n + 1; # this walks the line intil the end...
            if ($n eq '19') {
                $n = '0';
            }
        }

        @List1 = split(/,/, $Final);
        if ($List1[16] eq $Hashnumber1)
        {
            system ('touch UserInfo.txt');
            open(USER, '>>UserInfo.txt') or die "File not found...\n";
            # could check $val1 to see if it's wacky... and drop it ???
            if(defined($val1)) {
                print USER "$Final, $val1 \n";
                #... print "$Final $val1 \n";
            } # end if
        }
    }
} # end while
# ok, you have the info now in UserInfo.txt, so let's clean it up and display it
close(RESORT);
close(USER);

```

```

                                FinalUPN02
open(USER, 'UserInfo.txt') or die "File not found...190...
UserInfo.txt...\n";
while(<USER>) {
    $Line = $_;
    chomp($Line);
    @Lin = split(/,/,$Line);
    if ($Lin[15] eq '3') {
        $Interface = $Lin[18];
    }
    if ($Lin[15] eq '4') {
        $Mac = $Lin[18];
    }
}
# Output section .....

#print "Interface is..... $Interface\n";
#print "MAC is..... $Mac\n";
#print "User IP address is. $xTargetIP\n";
#print "Switch IP is..... $xSwitchIP\n";
#print "Protocol is..... $Proto\n";
#print "Port Number is.... $Port\n";
#print "\n";
#print "End of Program...\n\n";

# Finally, we write this to a log file with a time/date stamp....

($Sec, $Min, $Hr, $DayOfMonth, $Month, $Yr, $Weekday, %DayofYr, %IsDST) =
    localtime(time);
$RealMonth = $Month + 1;
$FullYr = $Yr + 1900;

open (LOG, '>>LogFile.txt') or die "Couldn't open LogFile... \n";
print LOG
"\n
Time-Date..... $Hr:$Min:$Sec $RealMonth-$DayOfMonth-$FullYr
User IP address is.... $xTargetIP
User MAC is..... $Mac
Switch IP is..... $xSwitchIP
Switch port is..... $Interface
Protocol is..... $Proto
Port number is..... $Port\n";

close (LOG);

# .....
# Section 3 SNMPset
# Here we use the Switch/Port data from section2 with the Proto/Port data
# from Dragon to build the discard rule and apply it to the switch/port. We will
# then update the log file...
# .....

# The process...
# 1) Create the discard valn
# 2) Create the rule (assign ports here as well!)
# 3) Enable rule...

# This is the clever secret decoder ring section .....
# Mapping the physical ports to their binary postions gives the following !!!
#
#      Hex      code      position      port#
#      80       \200      10000000      1
#      40       \@        01000000      2
#      20       \"blank"   00100000      3
#      10       \020      00010000      4

```



```

#           08      \b           00001000      5
#           04      \4           00000100      6
#           02      \2           00000010      7
#           01      \1           00000001      8
#           00      \0           00000000      null
#
# Use these value to set the IngressPortList values.....
# Now, how do we map the 'ifs' number into the code????
# what if we divided the number by 8, with the remainder = the code
# and the leading port groups (8) get a zero....
# Also, this will over-write what ever was there, so we might need to be
# able to read the current values and save them...? If so, that would make the
# code(s) messy...

```

```

use Net::SNMP;
use vars qw($session $error $result $classvar $vlanclasslist);

```

```

($session, $error) = Net::SNMP->session(
-hostname      => $xSwitchIP,
-community     => $xCommString,
-port         => 161);

if (defined($session)) {
printf("Error1 on session creation: %s.\n", $error);
exit 1; }

```

```

# We will need to setup the 'instance' and 'value' vars to feed into the
# We need to read the $Proto and then load the correct $Protox value...

```

```

# Enterasys Classification Numbers
# .....
#01      etherType(1)
#02      llcDsap$sap(2)
#03      ipTypeOfService(3)
#04      ipProtocolType(4)
#05      ipxClaSSOfService(5)
#06      ipxPacketType(6)
#07      ipAddressSource(7)
#08      ipAddressDestination(8)
#09      ipAddressBilateral(9)
#10      ipxNetworkSource(10)
#11      ipxNetworkDestination(11)
#12      ipxNetworkBilateral(12)
#13      ipUdpPortSource(13)
#14      ipUdpPortDestination(14)
#15      ipUdpPortBilateral(15)
#16      ipTcpPortSource(16)
#17      ipTcpPortDestination(17)
#18      ipTcpPortBilateral(18)
#19      ipxSocketSource(19)
#20      ipxSocketDestination(20)
#21      ipxSocketBilateral(21)
#22      macAddressSource(22)
#23      macAddressDestination(23)
#24      macAddressBilateral(24)
# .....

```

```

if ($xProto eq TCP) {$Protox = '18';}
elsif ($xProto eq UDP) {$Protox = '15';}

```

```

$vlan = '4000'; # This is the default discard VID, we could change this...
$port2 = '0'; # End of range port number... 0 if not used...
$vlanclass = "1.3.6.1.4.1.52.4.1.2.16.6.1.4.1.5.$vlan.$Protox.$Port.$Port2";

```

# FinalUPN02

```

$vlancreateOID = "1.3.6.1.2.1.17.7.1.4.3.1.1.$vlan";
$vlanName = "Discard";
$VlanDiscard = $session->set_request ($vlancreateOID, OCTET_STRING, $vlanName);
# print "New discard vlan info... \n $vlancreateOID and $vlanName\n";
$enableDiscardOID = "1.3.6.1.2.1.17.7.1.4.3.1.5.$vlan";
$VlanDiscardEnable = $session->set_request ($enableDiscardOID, INTEGER, 1);

# $vlanclassList = '1.3.6.1.4.1.52.4.1.2.16.6.1.4.1.5.4000.18.80.0';
# Here is where we need to convert the ifs number $Interface into a set code...
# Use the divide by 8 and all will be well...
# $classvar will be of a different length for different switches...! 24/36/48
# And then there are the ISLs... how to filter these out... 52.4.1.3.7.1.1.4.3.1.1

$block = int($Interface / 8); #Gives the rounded down integer
$rem = ($Interface % 8); #Gives the remainder
# Notice that the active block is at the [block+1] position !

# Begin the if this port then that code section...
if ($rem == 1) {$remvar = "\200";}
elseif ($rem == 2) {$remvar = "@";}
elseif ($rem == 3) {$remvar = " ";}
elseif ($rem == 4) {$remvar = "\020";}
elseif ($rem == 5) {$remvar = "b";}
elseif ($rem == 6) {$remvar = "4";}
elseif ($rem == 7) {$remvar = "2";}
elseif ($rem == 0) {$remvar = "\1";}

# blank block filler... meaning that the value is zero.
$f = "\0";
# Need to correct for a zero remainder when on a bit boundry...

#.....
if ($rem == 0) {$block = $block - 1;

if ($block == 0) {$classvar = "$remvar";}
elseif ($block == 1) {$classvar = "$f$remvar";}
elseif ($block == 2) {$classvar = "$f$f$remvar";}
elseif ($block == 3) {$classvar = "$f$f$f$remvar";}
elseif ($block == 4) {$classvar = "$f$f$f$f$remvar";}
elseif ($block == 5) {$classvar = "$f$f$f$f$f$remvar";}
} # end if

if ($block == 0) {$classvar = "$remvar";}
elseif ($block == 1) {$classvar = "$f$remvar";}
elseif ($block == 2) {$classvar = "$f$f$remvar";}
elseif ($block == 3) {$classvar = "$f$f$f$remvar";}
elseif ($block == 4) {$classvar = "$f$f$f$f$remvar";}
elseif ($block == 5) {$classvar = "$f$f$f$f$f$remvar";}
} # end else reloop the above code block...

#.....

#...print "Blank blocks are...$block and the port is...$rem\n";
#...print "ctVlanClassifyIngressList Value is...\n$vlanclass\n";

#...print "Testvars... $vlan, $Protox, $Port, $Port2 \n";
$result = $session->set_request (
    $vlanclass, OCTET_STRING, $classvar);
$ROWOID = "1.3.6.1.4.1.52.4.1.2.16.6.1.4.1.6.$vlan.$Protox.$Port.$Port2";
$rowStatus = $session->set_request ($ROWOID, INTEGER, 1);

```

# FinalUPN02

```
if (!defined($result)) {
printf("Error3 on result in vlanclass set: %s.\n", $session->error);
$session->close;
exit 1; }

# If no errors, then the filter was set, so update the log file for Dragon...
open (DRAGONLOG, '>>DragonLog.txt') or die "Couldn't open LogFile... \n";
print DRAGONLOG
"$Hr:$Min:$Sec $RealMonth-$DayofMonth-$FullYr: $xTargetIP: $DestIP: $Mac:
$xSwitchIP: $Interface: $Proto: $Port: $EventName: $Sensor: $Response\n";
close (DRAGONLOG);

$session->close;

exit 0;
```